1   **IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

2

3

4   APPLICATION FOR UNITED STATES PATENT

5

6   FOR

7   **MANAGING A DISTRIBUTED DIRECTORY DATABASE**

8

9

10

11

12

13   Inventors: Sriram Krishnan, Andreas L. Bauer, Gregory W. Lazar

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31   Attorneys:

32   Joel Wall, Esquire

33   P.O. Box 169

34   Hopkinton, MA 01748

35   508-435-4432

36

37   September 25, 2001

38

39

1

2 # MANAGING A DISTRIBUTED DIRECTORY DATABASE

3

4 ## CROSS REFERENCE TO RELATED APPLICATIONS

5 This patent application relates to another patent application filed on even date

6 herewith having co-inventors Sriram Krishnan and Gregory W. Lazar, assignee in

7 common with this patent application, the title of "Resolving Multiple Master Node

8 Conflict in a Distributed Directory Database", and which is incorporated by reference

9 herein in its entirety.

10

11 ## BACKGROUND OF THE INVENTION

12 **1. Field of the Invention:**

13 The present invention relates to apparatus, methodology, systems, and/or

14 computer program product for managing a directory database (DDB) and, more

15 particularly, relates to managing a distributed DDB within a computer network

16 environment such as a client server network.

17

18 **2. Description of Prior Art:**

19 As may be observed by even the most casual computer system user, computer

20 networks appear to be continuously evolving into ever-more sophisticated

21 communication systems. Accordingly, computer networks are having an ever-increasing

22 impact on modern-day lifestyle for many people, as the Internet is demonstrating.

23 Computer networks include local area networks (LANs) such as might be encountered

1   within a singular and localized corporate business organization where corporate functions

2   (engineering, marketing, sales, advertising, human resources, etc.) are linked together in a

3   localized computer network. Computer networks also include wide area networks

4   (WANs) such as might be encountered within a corporate business organization that is

5   nationwide or even worldwide in scope. The Internet is the premier example of a WAN,

6   but is one that is obviously not constrained within any particular business organization

7   and allows access to virtually any organization or individual. It is widely understood that

8   these networks not only utilize hard-wire bus communication paths to accomplish their

9   objectives, but can also utilize wireless telecommunication links via satellites and the like

10   as well.

11

12   Computer network configurations usable within LANs and WANs include client-

13   server network configurations. A client or workstation (typically having a user interface)

14   can be networked to multiple servers which serve the client in a variety of ways. Such a

15   client-server computer network configuration can be particularly useful in certain

16   applications, such as in managing a computer data storage system. A client-server

17   network can be used with the Internet. Thus, quality of operation of a storage system

18   within a client-server network operatively coupled over the Internet not only can impact

19   quality of performance of that client-server computer network, but can also impact

20   quality of service received by multiple human users connected through the Internet to that

21   client-server network. A failed or degraded storage system thus can escalate into a failed

22   or degraded service for multiple human users.

23

1   A client-server network today may typically be based on an object oriented

2   computer system which means that such system employs one or more object-oriented

3   computer languages such as C++, XML (eXtensible Markup Language), JAVA, and/or

4   others. Briefly, an object, in computer software terms, also referred to as a "node", is a

5   dedicated area of memory which can be thought of as an impervious container holding

6   both data and instructions within itself, both defining itself and its relationships to other

7   objects in the computer system or network. Such object or node can send and receive

8   messages to and from other objects, respond and react to such messages (e.g. commands)

9   but shall normally be impervious to internal scrutiny. For example, in the above-noted

10   computer data storage system (a kind of computer) each object (system object) may

11   describe or relate to a specific tangible detail in the storage system or in the storage

12   system's processor (e.g., details such as those describing or relating to aspects of

13   operation of the processor's cooling-fan, power switch, cache memory, power supply,

14   disk drive interface, etc.). These tangible objects (nodes) in the storage system can send

15   messages to each other within the storage system and to other objects outside the storage

16   system over the network with which they are operatively coupled. Also, the storage

17   system itself can be an object and interact as a node with other nodes in a network.

18

19   The relationship between and amongst these specific objects in the storage system

20   is usually visualized or characterized as a "tree" of objects. In a tree, each such object

21   hangs off a preceding object as if in a parent-child or inheritance relationship, with many

22   children hanging from a parent not being an atypical configuration. In addition to these

23   tangible kinds of objects, logical units (LUNs) are other nodes or objects that can be

1   contained within the tree. For example, a storage system object can have several LUN

2   objects as its children which, in turn, can have various disk objects as their children, etc.

3   These kinds of objects are generically referred to herein as "system objects" since they all

4   relate to a system or to components within a system, whether it is a storage system,

5   computer system, disk drive system, or some other system, and representations of these

6   objects can typically be displayed on a computer terminal's graphical user interface

7   (GUI) in this tree fashion. However, in contrast, other kinds of objects (nodes) can also

8   be formulated which do not relate to a system or its components per se, such as objects

9   relating to user actions and represented on the GUI in other ways. (User actions are any

10   commands or operations initiated by the user, such as, for example, creating a LUN or

11   downloading new software to a disk, etc.). In addition, there can be yet other kinds of

12   network nodes beyond the two types mentioned such as nodes comprising or represented

13   by a communication tree.

14

15   Accordingly, it shall be appreciated that there can be a very large number of

16   nodes of various kinds or "personalities" to keep track of and manage within even a

17   relatively small and local computer network. Any one or more of these nodes can fail,

18   for one reason or another, either temporarily or permanently, and certain other nodes can

19   be added or removed by network users under certain conditions, and this complex and

20   dynamic network node scenario must be efficiently and effectively managed if the

21   computer network's intended purpose is to be fulfilled. Moreover, this network node

22   management problem is compounded when the network is large, and possibly worldwide.

23   The subject of these computer network nodes is discussed further in two patent

1   applications filed by the assignee of the present invention: "Plug and Play Interface for

2   User Actions", Desai et al, U.S. Serial No. 09/916102, filed July 26, 2001 and "Scalable

3   Communication Within a Distributed System Using Dynamic Communication Trees",

4   Bauer et al, U.S. Serial No. 09/877862, filed June 8, 2001, both of which are incorporated

5   by reference herein in their respective entireties.

6

7   One prior art solution to the problem of managing network nodes is to elect one

8   node to have complete information about all other network nodes in the system.

9   Similarly, a group of nodes can be elected where each elected node in the group is

10   assigned to its own subnet or domain of nodes and has complete information about all

11   other nodes in its own subnet or domain. A subnet or domain is a network unto itself.

12   All other nodes in that network, or in that subnet as the case may be, then seek

13   information necessary to their functioning from their respective one elected node and are

14   thus managed through that one elected node. The inherent weakness in this configuration

15   is that if such one elected node fails, then no other node in its respective network or

16   subnet can function resulting in a failed network or subnet. This is a single point of

17   failure design which is not an optimum design because of at least this problem.

18

19   Prior art attempts to make this single point of failure design more reliable

20   included use of backup nodes, to take over functioning of an elected node if and when the

21   elected node failed. The problem with this backup design is that backup nodes can also

22   fail and then the result is the same as before – a failed network or subnet within the

23   network. Furthermore, when dealing with a worldwide network, which is not atypical

1   today, if backup nodes are being relied upon because of potential elected node failures,

2   and if all backup nodes are being maintained in one locale (e.g. United States of

3   America) for convenience, security, or other purposes, then virtually all network node

4   information can be destroyed for that global network if a disaster at the backup locale

5   destroys the backup nodes. This would be a major disruption. All worldwide users

6   (Europeans, Asians, etc.) of that worldwide network would thus lose the service provided

7   by that network.

8

9   Thus, there is a need for an improved technique for managing nodes in a network,

10   whether a large or small network, whether a worldwide or localized network, whether a

11   client-server or otherwise-configured network, and embodiments of the present invention

12   provide this improved technique - a welcome solution to these problems and

13   shortcomings of the prior art.

14

15   **SUMMARY OF THE INVENTION**

16   Embodiments of the present invention include apparatus, method, system, and/or

17   computer program product for managing a directory data base (DDB) distributed

18   throughout a plurality of nodes in a computer network configuration. A DDB is a

19   database of directory information, such as a directory of addresses of these nodes. These

20   embodiments operate within such computer network having a plurality of such nodes.

21   The nodes interact with computer network information. The nodes can receive, store,

22   modify and/or transmit the information. These embodiments manage the nodes by

23   establishing a DDB in each of the nodes and controlling contents of each DDB to be

1   substantially identical to contents of every other DDB. In one aspect of the present

2   invention, there is apparatus, method, system, and/or computer program product having

3   program code that manages the nodes in a manner that establishes a DDB in each of the

4   nodes and controls contents of each DDB to be consistent, or substantially identical to

5   contents of every other DDB, while avoiding a single point of failure.

6

7   In another aspect of the present invention, the computer network information is

8   computer data and domain configuration status information and each of the nodes has a

9   unique internet protocol (IP) address. Each unique address is associated with its

10   respective node thereby providing an IP-address-respective-node association. The

11   association for each of the nodes is combined into a network IP association. The network

12   IP association is distributed to the DDB in each of the nodes. The most current domain

13   configuration status is maintained in the DDB of each of the nodes. One of the nodes is

14   selected as a master node. All of the other nodes are subordinated to the master node in a

15   configuration defined by the master node and all of the other nodes. A change to the

16   domain configuration status is responded to by the master node in a manner to maintain

17   the contents of each DDB substantially identical to the contents in every other DDB.

18

19   In yet another aspect of the present invention, in a computer network having a

20   plurality of nodes each having a DDB, one of the nodes is the master node. The master

21   node is used to maintain contents of the DDB in each of the nodes consistent throughout

22   the plurality. If the master node fails, another of the nodes in the plurality is selected as

23   the new master node. The new master node can be selected by the global administrator

1   by invoking a select master dialog in a graphical user interface (GUI). The replacement

2   node can be selected from the configuration. Failed and potentially failed nodes are

3   detected and each is subjected to pinging until each one recovers. Failed nodes include

4   the failed master node. The contents of the DDB in recovered nodes are updated to

5   match the contents of the DDB in the new master node including the new master node's

6   DDB version number. There is a handshaking protocol which is followed between the

7   new master node and the recovered node when it is updated.

8

9   In still yet another aspect of the present invention, changes to domain

10   configuration status include adding a first node to the plurality of nodes, deleting a

11   second node from the plurality of nodes, the failing of a third node in the plurality of

12   nodes and/or the failing of a network link between the master node and a fourth node in

13   the plurality of nodes. The master node responds to any one or more of these changes as

14   follows:

15   (a) Node Addition: The global administrator utilizes the GUI to add a node to the

16   configuration. The adding of the first node is handled by determining if it is being added

17   through the master node to obtain a "master-added node" or through one of the other

18   nodes acting as a portal node to obtain a "portal-added node" (non-master-added node).

19   Under a master-added node condition, the DDB in the master node is updated with the IP

20   address of the first node, and the first node is informed that its master is the master node;

21   the IP address of the master node is entered in the DDB of the first node which

22   acknowledges the master node; and, the IP address of the first node is sent as an update to

23   all other nodes in the configuration by the master node. By contrast, under a portal-added

1   node condition, a cache memory in the portal node holds the IP address of the first node;

2   a node to master handshake is performed between the portal node and the master node;

3   the portal node informs the master node of the IP address of the first node; the master

4   node updates its DDB with the IP address of the first node, and informs the first node that

5   the first node's master is the master node; the first node enters in its DDB the IP address

6   of the master node and acknowledges the master node; and, the master node sends the IP

7   address of the first node as an update to all other nodes in the configuration. Whether or

8   not the first node is added through the master node, a master to node handshake is

9   undertaken between the master node on the one hand and both all of the nodes in the

10   configuration and the first node on the other hand, as part of the update.

11   (b) Node Deletion: The global administrator utilizes the GUI to delete a node

12   from the configuration. The deleting of the second node is handled by determining if it is

13   being removed through the master node. If so: the DDB in the master node is updated by

14   removing the IP address of the second node from the master node's DDB; the second

15   node is informed that it is no longer included in the configuration and it is detached from

16   the configuration; all contents of the second node's DDB is erased; and, an update is

17   sent to all remaining nodes in the configuration. If not: a "portal-removal node" other

18   than the master node is selected through which the second node is removed from the

19   configuration; a cache memory is included in the portal-removal node; the IP address of

20   the second node is stored in the cache memory; a node to master handshake is performed

21   between the portal-removal node and the master node; the master node is informed to

22   remove the IP address of the second node from the master node's DDB; the DDB in the

23   master node is updated by removing the IP address of the second node from the master

1   node's DDB; the second node is informed that it is no longer included in the

2   configuration and it is detached from the configuration; all contents of the second node's

3   DDB is erased; and, an update is sent to all remaining nodes in the configuration.

4   Whether or not the second node is removed through the master node, a master to node

5   handshake is undertaken between the master node and all remaining nodes in the

6   configuration as part of the update sent to all remaining nodes in the configuration.

7   (c) Node Failure: The failing of the third node in the plurality of nodes can occur

8   under conditions in which the master node is known to the third node, or under other

9   conditions in which the master is unknown to the third node (third node failing while

10   being added to configuration, and/or, master node replaced during time of failure of third

11   node). If failing occurs under known master conditions: version numbers are established

12   to identify versions of the DDB in each of the plurality of nodes; the master is

13   continuously polled by all other nodes in the plurality at regular intervals to obtain the

14   most current one of the version numbers of the DDB in the master node; after the third

15   node recovers the master node responds to the polling received from the third node and

16   sends the most current one of its version numbers to the third node; and, the DDB in the

17   third node is updated if the most current one of the version numbers sent by the master

18   node does not match the version number of the DDB in the third node. By contrast, if

19   failing occurs under unknown master conditions: the master node repetitively pings the

20   third node at predetermined intervals until the third node recovers and sends a recovery

21   signal to the master node; and, the master node responds to the recovery signal and

22   updates the DDB in the third node as may be needed. Under either condition, as part of

23   the update, handshaking is undertaken between the master node and the third node.

1   (d) Link Failure: The master node handles a network link failing: version

2   numbers are established to identify versions of the DDB in each of the plurality of nodes;

3   the master is continuously polled by all other nodes in the plurality at regular intervals to

4   obtain the most current one of the version numbers of the DDB in the master node; after

5   recovery of the failed network link, the master node responds to the polling received from

6   the fourth node over the network and sends the most current one of its version numbers to

7   the fourth node; and, the DDB in the fourth node is updated if the most current one of the

8   version numbers sent by the master node does not match the version number of the DDB

9   in the fourth node; handshaking between the master node and the fourth node is

10   undertaken as part of the updating.

11

12   In a further feature of the present invention, node to master handshaking is

13   undertaken by an inquiring node in a plurality or configuration of nodes of its master

14   node under certain circumstances. The address of a first node in the plurality that is

15   presumed by the inquiring node to be the inquiring node's master is obtained. A

16   determination is made from that first node if that first node is the inquiring node's master

17   node. If yes, the handshaking is completed. If not, then an inquiry is made of that first

18   node regarding who is the new master for the inquiring node. If the new master is thus

19   determined, its address is provided to the inquiring node and the handshaking is

20   completed. If the new master is not thus determined, then a decision is made whether or

21   not to request the global administrator to configure information identifying the new

22   master for the inquiring node. If such request is made of the global administrator, such

23   information is provided to the inquiring node and the handshaking is completed. If such

1   request is not made of the global administrator the inquiry of the first node is repeated

2   including the repeating of any succeeding activity based on results of that repeated first

3   node inquiry until the handshaking is concluded.

4

5   In a still further feature of the present invention, master to node handshaking is

6   undertaken by a master node in a computer network configuration having a plurality of

7   nodes including the master node, each of which has a DDB. Contents of the DDB

8   include its respective DDB version number. The master node is used to maintain the

9   contents in each of the plurality of nodes consistent throughout the plurality. The

10   handshaking is initiated as a function of the master node undertaking to provide an update

11   message including the master node's address to all other of the nodes in the plurality in

12   response to a change to the network configuration. The DDB in each of the other of the

13   plurality of nodes has an address of a purported master node. The handshaking

14   comprises, for each node, determining if the master node address in the update message

15   matches the address of the purported master node. If no match, the update message is

16   rejected and the handshake is concluded. But if there is a match, then a determination is

17   made with regard to a match between the version number of contents of the DDB in the

18   node and the version number of contents of the DDB in the master node *before* the update

19   message. If no version number match, the update message is accepted into the DDB of

20   the node and contents of the node's DDB is replaced with contents of the master node's

21   DDB. But if there is a version number match, only that portion of the update message is

22   accepted into the node's DDB which is different from its contents and the handshaking is

23   completed.

1

2   It is thus advantageous to employ embodiments of the present invention in client-

3   server networks and any other networks where management of a distributed directory

4   database is required. In various embodiments of the present invention summarized

5   herein, there is no single point of failure since each DDB contains all information of

6   every other DDB (all other DDBs). If a master node fails, the global administrator

7   simply appoints a different node, such as another one from the configuration, to be

8   operative as new master. If nodes or links fail the polling and pinging features keep track

9   of these failures and become self correcting in due course, unless a failure is permanent,

10   in which case nothing short of physical replacement of failed components can help. At

11   least because of avoidance of a single point of failure architectural design, there cannot

12   be catastrophic failure scenarios while using embodiments of the present invention

13   because of failed master or other nodes, as contrasted with likely occurrence of such

14   scenarios while using prior art node management architectures .

15

16   It is therefore a general object of the present invention to provide an improved

17   computer network.

18

19   It is another general object of the present invention to provide improved

20   distributed management software.

21

1   It is a further general object of the present invention to provide an improved

2   technique for managing a distributed directory database in a computer network such as a

3   client server network.

4

5   It is a still further general object of the present invention to provide an improved

6   technique for managing a distributed directory database in a computer network in a

7   manner that avoids a single point of failure design and maintains the content in the

8   distributed directory database consistent throughout the network, regardless of node or

9   related failures, node additions and/or node deletions.

10

11   It is an even further object of the present invention to provide an improved

12   computer data storage system.

13

14   Other objects and advantages will be understood after referring to the detailed

15   description of the preferred embodiments and to the appended drawings wherein:

16

17   **BRIEF DESCRIPTION OF THE DRAWINGS**

18   Fig. 1 is a block diagram of a client server network having multiple servers and a

19   network node cloud in which embodiments of the present invention are particularly

20   useful;

21   Fig. 2 is a schematic diagram showing unconfigured nodes within a defined

22   boundary;

1   Fig. 3 is a schematic diagram showing configured nodes in three different

2   domains within the same defined boundary;

3   Fig. 4A is a schematic diagram of a network containing two domains of network

4   nodes having a master node in each domain;

5   Fig. 4B is a schematic diagram of the directory database distributed throughout

6   one domain in the network of Fig. 4A;

7   Fig. 5 is a flowchart reflecting an algorithm executed by operation of

8   embodiments of the present invention in establishing a configuration;

9   Fig. 6 is a flowchart reflecting an algorithm executed by operation of

10   embodiments of the present invention in adding a node to a configuration;

11   Fig. 7 is a flowchart reflecting an algorithm executed by operation of

12   embodiments of the present invention in removing a node from a configuration;

13   Fig. 8 is a flowchart reflecting an algorithm executed by operation of

14   embodiments of the present invention in performing a master to node handshake;

15   Fig. 8A is a flowchart reflecting an algorithm executed by operation of alternative

16   embodiments of the present invention in performing a master to node handshake;

17   Fig. 9 is a flowchart reflecting an algorithm executed by operation of

18   embodiments of the present invention in performing a node to master handshake;

19   Fig. 10A is a flowchart reflecting an algorithm executed by operation of

20   embodiments of the present invention in responding to a failed network node that does

21   not know which node is its master node;

1   Fig. 10B is a flowchart reflecting an algorithm executed by operation of

2   embodiments of the present invention in responding to a failed network node that does

3   know which node is its master node;

4   Fig. 11 is a flowchart reflecting an algorithm executed by operation of

5   embodiments of the present invention in responding to a failed network link;

6   Fig. 12 is a flowchart reflecting an algorithm executed by operation of

7   embodiments of the present invention in responding to a failed master node;

8   Fig. 13 is a representation of a typical computer terminal screenshot associated

9   with certain embodiments of the present invention employing a GUI, the screenshot

10 reflecting a dialog box of the type which would enable a global administrator user to

11 control master node selection; and,

12   Fig. 14 is a representation of a typical computer terminal screenshot associated

13 with certain embodiments of the present invention employing a GUI, the screenshot

14 reflecting a dialog box of the type which would enable a global administrator user to

15 control domain configuration or re-configuration.

16

17   **DESCRIPTION OF THE PREFERRED EMBODIMENTS**

18   Preliminarily, section headings which refer to figures to which they principally

19 relate are used hereinbelow as an organizational aid. However, there may be discussion

20 of subject matter reflected in a particular figure which appears in a section headed by a

21 figure number associated principally with other subject matter. The section headings are

22 thus not intended to be construed in a limiting manner. The terms "master node" and

23 "master" may be used interchangeably herein and have the same meaning. And, it should

1   be kept in mind that any network in which embodiments of the present invention are

2   operative has an intended purpose beyond management of its nodes. For example, in a

3   computer data storage system network, in which embodiments of the present invention

4   are particularly useful, the main purpose of each node in such network is to enhance the

5   overall network function of storing data, and management of its distributed DDB is only

6   one operation involved in that overall data storage function.

7

8   **Figure 1 – Client-Server Network**

9   Fig. 1 is a block diagram of an exemplary client server network having network

10   components such as multiple servers and a network node cloud. Embodiments of the

11   present invention are particularly useful in this network. User interface (UI) or client or

12   management workstation 100 is connected by bidirectional bus structures 106 and 107 to

13   servers 101, 102, and 103. UI 100 also is operatively coupled to network node cloud 105

14   by bidirectional bus structures 106, 107, 108. And server 103 is operatively coupled to

15   storage system 104 by bidirectional bus 109. The network is not limited to three servers

16   or to one storage system, and more than three servers are intended to be suggested by the

17   broken line representing bus 107 in connection with the dots between servers 102 and

18   103. In a particular configuration, UI 100 can be a Windows NT or UNIX workstation

19   that runs Navisphere 6.X user interface software or equivalents thereof; servers 101-103

20   can be any Windows NT machines or EMC model FC 4700 storage systems running

21   Navisphere 6.X management servers or equivalents thereof; and, storage system 104 can

22   be an EMC model FC 4700 storage system or its equivalent. Network node cloud 105 is

1   intended to represent a network such as a network of EMC FC-4700 storage systems or

2   their equivalents, containing a huge number of nodes, possibly thousands or more

3

4   As shown in Fig. 1, a directory data base (DDB) is situated in each network

5   component shown in the network. A DDB is a database of directory information

6   reflecting identities and addresses of both the node in which it resides and the other nodes

7   with which it communicates, much like a telephone directory book can list names, phone

8   numbers and addresses of all telephone subscribers to a telephone network. Since each

9   network component may include a large number of network nodes, as discussed earlier,

10   the DDB designation shown in each component is intended to represent DDBs distributed

11   throughout all nodes in that component. (The DDB shown in server 101, for example,

12   represents a plurality of DDBs.) In accordance with principles of the present invention to

13   be discussed in more detail below, under certain circumstances the DDBs in each

14   component are maintained consistently throughout the entire network. (The entire

15   network could be divided into independent subnets or domains, each being a sub-

16   network, described below.) In other words, the information content in any one particular

17   DDB in any particular node of any particular component in the network is maintained

18   consistent (replicated, duplicated, or made substantially identical) with content of any

19   other DDB in the network. Using the telephone book analogy, each telephone user

20   within that calling locale (the network) should be given the same revision or version of

21   the telephone book (the same DDB content – the same version number). Then, all such

22   users (nodes) have current information about all other telephone users listed in their

23   respective books. If a telephone user moves away, (node deleted) or if a new telephone

1   user moves into the locale (node added), then the telephone book needs to be updated

2   with the new information. The new telephone user needs to be given a complete and

3   current telephone book (current DDB). Under these conditions all telephone users in that

4   locale can rely on their respective telephone directories and can properly function.

5   Similarly, if the information content in each DDB in each node in the network is

6   maintained consistent regardless of network node changes (node additions, node

7   deletions, node failures, master node failures, failed links, etc.), then the computer

8   network nodes can rely on their respective database directories (DDBs) and can properly

9   function.

10

Figures 2 & 3 – Nodes

12   Fig. 2 is a schematic diagram showing unconfigured nodes within a defined

13   boundary. Bounded space 200 may represent, for example, a portion of a computer

14   system which has not yet been organized into an interrelated or cooperative entity, such

15   as may exist in a recently purchased system just delivered to the purchaser and prior to

16   such system being configured. The squares in the space represent nodes. The arrows in

17   the squares pointing in random directions are intended to suggest that these nodes are not

18   yet aligned with each other in a cooperative computer network effort. Each node's DDB

19   has only its own identification information in it – its own IP address. Thus, each node in

20   this scenario is functionally unaware of other nodes within this boundary and if it were to

21   respond to any computer information stimuli (data and/or commands) such response

22   would probably be independent of, uncooperative with, and/or contradictory to responses

1   to the same stimuli received by other unaligned nodes. Bounded space 200 is not a

2   network - yet.

3

4   Fig. 3 is a schematic diagram showing the same nodes as those in Fig. 2, but now

5   shown as configured nodes in a network of three different domains, 301, 302, and 303

6   within the same defined boundary as that of Fig. 2. The same bounded space 200 thus

7   contains the same nodes as before, but which have now been aligned, figuratively

8   speaking, as shown by arrows, and segregated into domains within the same network

9   boundary. (There is no literal, physical alignment or orientation of nodes into a particular

10   geometrical direction – Fig. 3 reflects such orientation as an aid to comprehension.)

11   Alignment, in this context, means that a particular node is subordinate to, and takes its

12   direction from, *only* its master node. In accordance with principles of the present

13   invention to be discussed in more detail below, in domain 301 nodes are aligned to node

14   M1, in domain 302 nodes are aligned to node M2, and in domain 303 nodes are aligned

15   to node M3. Nodes M1, M2, and M3 are termed "master nodes". The master nodes are

16   nodes selected from the prior field of nodes within bounded space 200 and are made a

17   little special or privileged as compared with the nodes now in alignment to them.

18   Aligned nodes are termed "participating nodes". The responsibility of a master node is to

19   ensure that contents of DDBs of all participating nodes in its configuration (all nodes in

20   domain 301 including master M1 are in the configuration of master M1, etc.) are

21   maintained consistent or substantially identical. The master node's job is to replicate or

22   duplicate the information content stored in its own DDB into each DDB in each

23   participating node in its configuration. As noted, after each such DDB is replicated as

1　　desired it can be conceptualized as being "aligned" to its master node, as exemplified by

2　　the arrows superimposed on the nodes in domains 301, 302, and 303 pointing,

3　　respectively, in alignment to master nodes or masters M1, M2, and M3.

4

5　　　　These master nodes are selected by a computer network user termed a "global

6　　administrator", a user who is privileged as compared with other users in that computer

7　　network. Only the global administrator has proper authorization to select or change

8　　master nodes, and make other network changes. Other users have rights limited to

9　　merely using the network, as is. The global administrator selects nodes M1, M2, and M3

10　　through a network user interface, e.g. UI 100 of Fig. 1, based on factors relevant and

11　　important to the administrator. For example, selection of master nodes could be based on

12　　proximate locations of such nodes to those intended to be in their respective

13　　configurations. Another example would be selection of a master node based on its

14　　functional relevance to the intended function of its configuration. The internet protocol

15　　(IP) addresses of the nodes to be selected as master nodes are known to the global

16　　administrator by way of the GUI, about which more shall be discussed hereinbelow.

17

18　　　　　　　　　　**Figures 4A/B– Network & Domains - Overview**

19　　　　Fig. 4A is a schematic diagram of an exemplary network containing two domains

20　　of network nodes having a master node in each domain. This diagram shall be used in an

21　　overview discussion involving creation of a configuration, and shall use the example of

22　　adding nodes to the configuration as an introduction to certain concepts of the present

23　　invention. More detail about these concepts as well as other necessary actions of

1   removing nodes and handling various failure scenarios shall be presented hereinbelow in

2   connection with other Figures.

3

4   Assume that network 400 initially was not configured as shown, but that it was

5   merely a boundary which contained nodes 1-8 inclusive in an unconfigured state, such as

6   reflected in Fig. 2. The first action undertaken is for the global administrator to pick a

7   node as master, and, in this example, node 1 is picked. Node 1 is selected by way of a

8   GUI (not shown in this Fig.) through a "select master" dialog box which appears on a

9   terminal screen of the GUI and which reflects IP addresses of nodes in this boundary. At

10   this point, the master node represents a configuration of one node - no other node in the

11   otherwise unconfigured boundary is aware that node 1 has been made master node.

12   Then, assume that the global administrator adds nodes 2 and 3 to the configuration, in

13   that order, by sequentially selecting such nodes' respective IP addresses from a

14   "configure domain" dialog box on the terminal screen of the GUI, by placing the terminal

15   screen cursor and clicking the mouse on "buttons" appearing in this dialog box which

16   cause these nodes to be sequentially added to the configuration. More about this

17   procedure will be presented in connection with Figs. 13 and 14 hereinbelow. At this

18   point, however, status of all nodes is: DDBs in nodes 1, 2, and 3 each have the same

19   contents, namely: IP addresses of nodes 1, 2, 3, and a flag that node 1 is master, and

20   nodes 4-8 remain unconfigured for the time being. Domain 401 has thus far developed

21   into a three-node configuration. The unconfigured nodes are further discussed

22   hereinbelow.

23

1   Backtracking momentarily, when the global administrator is adding node 2 to the

2   configuration, such node is necessarily being added through, or by way of, the only node

3   that pre-existed node 2 in the domain 401 configuration, namely node 1. Accordingly,

4   node 1, acting as master node, forwards information to just-arrived node 2 which advises

5   node 2 that node 1 is going to be master for node 2 and that node 2 has been added to the

6   configuration of node 1. Node 2 is thus a "master-added node". Node 2 then checks the

7   version number of the DDB in information received from the master node. A version

8   number is a unique worldwide number (WWN), a number that if once used is never used

9   again. In this example of a new configuration being created out of a boundary of chaotic

10   or unaligned nodes the version number of the master node's DDB could be zero. If the

11   version number in information sent by the master node is different from the pre-existing

12   version number of the DDB in node 2 (the likely case), then node 2 needs to get a copy of

13   the entire contents of the master's current DDB and to update (replace) the DDB in node

14   2 to be identical to the DDB in node 1. But, if version numbers of the DDBs in nodes 1

15   and 2 are the same (not likely), then the foregoing update/replacement is avoided. The

16   same basic procedure is undertaken for node 3 and for any other node to be added to the

17   domain 401 configuration, but there is some significant variation to the procedure as a

18   function of which node in the configuration is selected by the global administrator to be a

19   "portal node"- the node through which a new node is to be added. If the portal node is

20   master node, then the above procedure is used, but if the portal node is a configured node

21   within domain 401 other than master node then a different procedure is used, as follows.

22

1   If the global administrator adds node 3 through node 2 thus making node 2 a

2   portal node for new node 3, node 2 stores the IP address of node 3 in a cache memory

3   within node 2 (not within the DDB of node 2), sends a message to master node 1 that

4   node 3 has been or is being added and provides the IP address of node 3 in that message.

5   If master node 1 successfully receives this message from node 2, master node 1

6   immediately updates itself by adding node 3 IP address information to its DDB and

7   changes its version number. The version number changes from one unique WWN to a

8   different unique WWN . Concurrently, node 1 sends a "success note" back to portal node

9   2 about which more will be said below. Further, master node 1 sends a message to node

10   3 that the master for node 3 is node 1 and also provides node 3 with the changed new

11   version number of node 1 as well as a copy of the contents of its entire DDB. Newly

12   arrived node 3 accepts such contents, updates its DDB therewith, applies the changed

13   new version number to such contents, and is thereafter a participating node along with

14   node 2 in a configuration designated as domain 401 in which node 1 is master node.

15

16   Although, node 2 has retained the IP address of newly-added node 3 in its cache

17   memory, node 2 has not yet been updated. The earlier-mentioned success note contains

18   at least the master node's prior version number as well as its changed new version

19   number. Node 2 compares the master node's prior version number received in the

20   success note against its own version number of its DDB. On the one hand, if these

21   version numbers match, portal node 2 updates its own DDB with node 3 IP address

22   information which is being stored in the node 2 cache memory, thereby relieving the

23   master node from unnecessarily returning node 3 IP address information back to node 2.

1   Portal node 2 also updates its version number to be the same as the changed new version

2   number of the master node which was received in the success note. But, on the other

3   hand, if the prior version number of master node 1 did not match the version number of

4   portal node 2, then portal node 2 destroys contents in its cache memory about new node 3

5   and waits for its update message from master node 1. (The update message is a change

6   notification to all participating nodes in the master node's domain and is part of the

7   master node's replication process.) If there were many other operative and participating

8   nodes in domain 401, they would all be subjected to this replication process and portal

9   node 2 would be just one of all such participating nodes being updated with information

10   about newly added node 3 at the time of the update message. A significant aspect of this

11   procedure is that portal node 2 receives minimal communication from master node 1 to

12   achieve update of its DDB as noted above. Accordingly, any master node is worked

13   minimally and network or configuration "housekeeping" traffic is kept minimal in the

14   present invention while maintaining contents of distributed DDBs consistent throughout

15   any given node configuration.

16

17   Next, consider node 4, another new node being added to the configuration of

18   domain 401. If the global administrator, by way of the computer network's GUI, selects

19   node 3 as portal node for node 4, and if node 3 cannot properly communicate with master

20   node 1 for one of a number of reasons such as the network link between master and node

21   3 (not shown in Fig. 4A) has failed, then node 4 cannot be added to the configuration

22   through node 3. In such an event, an error message is sent to the GUI so that the global

23   administrator knows to select another node as portal node for node 4. The global

1   administrator can again select node 2 as portal node, this time for node 4, and node 4 can

2   thereby be successfully added to the domain 401 configuration in a manner similar to the

3   adding of node 3. (Another reason why node 4 might not be able to be added to the

4   configuration is that the master node itself has failed in which case an error message is

5   again returned to the global administrator as before, but which additionally advises of

6   failure of a master node. In this case the configuration is frozen and cannot be changed

7   until a new master is created by the global administrator, and discussion of this event is

8   presented hereinbelow.)

9

10   Domain 402 is shown with nodes 5, 6, 7 and 8. These nodes are shown in a

11   configuration with node 6 as its master node. These four nodes in this configuration have

12   been created in the same manner as those in domain 401, but either by a different global

13   administrator from the one in control of domain 401, or by the same global administrator

14   intending to establish a different and mutually- isolated domain from domain 401.

15   Accordingly, these two domains represent two different configurations which can co-

16   exist in a single computer network without either domain interfering with the other.

17   More than two domains can co-exist in a single network and only two are shown as an

18   example. (Related subjects are further discussed hereinbelow in connection with the

19   section on "Marriage of Two Subnets"). Nodes in this domain are shown to be

20   interconnected by communication lines 403-408 inclusive in a particular manner, about

21   which more shall be said hereinbelow in connection with master node "pinging" and

22   participating node "polling" techniques.

23

1   In Fig. 4B, the DDB which is distributed throughout the configuration of domain

2   401 is depicted in schematic format. Content of this DDB reflects successful addition of

3   node 4 to the domain 401 configuration and is the same content for each DDB within

4   each of nodes 1-4. This DDB associates each of the nodes in the configuration with its

5   respective IP address. These nodes could thus be localized, as in a LAN, or could be far-

6   flung and located around the globe being connected through the Internet. Node 1 has a

7   flag associated with it, shown in the diagram as an "M" to designate node 1 as the

8   configuration's master. Since each node in the domain 401 configuration has these

9   contents in its DDB, each node knows that node 1 is its master, because each DDB

10   contains this "M" flag next to or associated with the node 1 address content in its

11   database. And each node in the configuration also knows the IP address of every other

12   node (all other nodes) in the configuration. The DDB contains domain identification

13   information, as reflected by "DOMAIN 401" shown in the Figure. And, the DDB also

14   contains a version number as shown, "DDB VERSION #__", to reflect or identify the

15   status or version of its content, namely, participating node information as currently stored

16   in its DDB. In the case of master node 1, as noted above, it has the capacity to forward to

17   participating nodes in its configuration not only its prior DDB version number which was

18   stored in its DDB, but also can forward its changed update version number as stored in its

19   updated DDB, in the event of a network configuration status change. A schematic

20   drawing for the DDB distributed throughout the configuration of domain 402 would be

21   similar to the DDB for domain 401, but would have different content reflecting different

22   IP addresses and version number.

23

1   The procedure of adding a new node to a configuration, whether a master or non-

2   master node in the configuration is used as portal node, requires certain handshaking

3   protocols. These shall be discussed in detail hereinbelow in connection with flowcharts

4   depicting their respective algorithms. Furthermore, as noted earlier, this discussion of

5   Fig. 4 has considered only the example of adding a new node in connection with a high

6   level overview of certain aspects of the present invention. Other changes to a

7   configuration including node deletion, node failure, master node failure, and master to

8   node link failure are considered in detail hereinbelow.

9

Figure 5 – Establishing Configuration

10

11   Fig. 5 is a flowchart reflecting an algorithm executed by operation of

12   embodiments of the present invention in establishing a configuration or domain. Initially,

13   in block 501 a chaotic array of nodes are defined within a boundary, such as those nodes

14   shown in Figure 2. The algorithmic process moves to block 502 wherein a node is made

15   master node by the global administrator through operation of the create master dialog box

16   in the GUI, thereby creating a configuration of one node. In this case, one of the nodes

17   within the boundary can be selected to be the master node. (A node outside of the

18   boundary could have been chosen by the global administrator to be the master node.)

19   Software, employed in preferred embodiments of the present invention, by which the

20   master is chosen and by which all other objectives of the present invention are

21   accomplished, is C++ object oriented software. Such software is the best mode now

22   known by which to implement the present invention, although other object oriented

23   software such as Java could have been chosen.

1

2   The algorithmic process then moves to block 503 where the global administrator

3   chooses other nodes through the GUI from either within or outside of the boundary to be

4   added to the configuration. Next, in block 504, the master node sends its IP address to all

5   the other nodes chosen to be within the configuration informing each of them that the

6   master node is their master. (Employing personification, in such "informing" the master

7   is effectively saying: "I am the master for you"!) This informing is also referred to herein

8   as the aforementioned pinging, which is also used to deal with certain failure scenarios,

9   and will be discussed in more detail in connection with Figs. 10A and 12. In block 505

10 each chosen node enters the master node's IP address in its DDB, acknowledges the

11 master node, and thus aligns itself to the master node. This acknowledgment is vital for

12 the master to receive from each node in its configuration so that the master knows that

13 each such node is properly aligned. Failure to receive an acknowledgment from a

14 particular node alerts the master that such node might have failed and sets up a repetitive

15 pinging activity. The actions in blocks 503, 504, and 505 are accomplished by way of the

16 configure domain dialog box in the GUI. The steps or actions in this Figure are repeated

17 as many times as are necessary to accomplish the total number of domains desired per

18 network. Thus, in Fig. 3, within network boundary 200, three subnets or domains are

19 formed by these steps being undertaken three times by either one or multiple global

20 administrators.

21

Figure 6 – Adding Node

1

2   Fig. 6 is a flowchart reflecting an algorithm executed by operation of

3   embodiments of the present invention in adding a node to a configuration or domain.

4   Discussion about adding a node was presented in connection with Figs. 4/4A above, and

5   is applicable here. Initially, in block 601, the global administrator decides to add a new

6   node to an existing configuration of nodes (either a subnet or an entire network), and

7   selects the new node by way of IP addresses shown in the configure domain dialog box in

8   the GUI (to be described in connection with Figs. 13 and 14). In block 602, the global

9   administrator chooses between adding the new node into the configuration by way of the

10   existing master node or by way of another node (a portal node) existing within the

11   configuration.

12

13   On the one hand, if the master node is chosen, then the algorithmic process moves

14   to the group of blocks 603 – 606. In block 603, the master node immediately updates its

15   own DDB with the new node's IP address since the new node is being processed through

16   the master node and the new node's IP address is available. Then, in block 604, the

17   master node informs the new node that it is the new node's master. Next, in block 605,

18   the new node enters the master node's IP address in its DDB and acknowledges its new

19   master. Thereafter, in block 606, the master sends an update notice to all nodes within its

20   configuration to advise them of the addition of the new node by providing the new node's

21   IP address for inclusion in their DDBs. The algorithmic process moves to Fig. 8 wherein

22   a master to node handshake is undertaken between the master node and each node in the

23   configuration as well as between the master node and the new node . In that handshake,

1   to be discussed in detail in connection with Fig. 8, contents of each node's DDB is

2   verified as being up-to-date or is otherwise is brought up-to-date resulting in consistent or

3   identical contents of all DDBs in the network.

4

5   On the other hand, if another node is chosen as a portal node by or through which

6   to add the new node, the algorithmic process moves from decision block 602 to block

7   607. In block 607 the other (portal) node holds the IP address of the new node in a cache

8   memory located in the portal node. Next, in block 608, a node to master handshake is

9   undertaken by the portal node with its master node to verify that such master node is, in

10 fact, the portal node's current master node. This handshake is undertaken in Fig. 9 and

11 shall be discussed in detail in connection with that figure. Thereafter, in block 609, the

12 portal node informs the master node of the new node's IP address. At this point, the

13 master node has the same information about the new node as it had earlier in the

14 discussion, when the decision made in block 602 was to choose the master node to

15 function as portal node. Thus, the algorithmic process now moves to that same point,

16 namely to block 603, and the above-described process for blocks 603 through 606

17 including the master to node handshake of Fig. 8 are repeated.

18

19   **Figure 7 – Removing Node**

20   Fig. 7 is a flowchart depicting an algorithm executed by operation of

21 embodiments of the present invention in removing a node from a configuration **or**

22 **domain**. Initially, the global administrator selects a node to be removed in block 701 by

23 again referring to the IP address information of all nodes in the configuration as shown in

1   the configure domain dialog box in the GUI (to be described in connection with Figs. 13

2   and 14). The global administrator makes a decision to remove the selected node through

3   either the master node or through another node in the configuration - a portal-removal

4   node.

5

6   On the one hand, if the global administrator decides to remove the selected node

7   through the master, the process moves to block 707 where the master node updates its

8   own DDB by removing the IP address of the selected node from its DDB. Then, in block

9   708, the master node informs the selected node that it has been removed from the

10   configuration and the selected node erases or destroys its own DDB and thereby detaches

11   itself from the configuration. Thereafter, in block 709, the master node sends an update

12   message to all nodes remaining within the configuration, and as part of that update the

13   master-to-node handshake depicted in Fig. 8 is performed.

14

15   On the other hand, if the global administrator decides to remove the

16   selected node through a portal-removal node, the process moves to block 703 wherein the

17   selected node can be removed either through itself as portal-removal node or through

18   another node in the configuration as portal-removal node. This choice is made by the

19   global administrator, but in either case the same process is followed. In block 704, the

20   chosen portal-removal node stores in its own cache memory the IP address of the selected

21   node. In block 705 a node to master handshake is performed as depicted in Fig. 9 to

22   ensure that the master node, as reflected by its IP address stored in the portal-removal

23   node's DDB, is the actual, current master node of the configuration. (Fig. 9 is discussed

1   in detail hereinbelow.) Thereafter, in block 706, the portal-removal node informs the

2   master node of the selected node's IP address to be removed from the master node's

3   DDB. At this point, the master node has the same information about the selected node as

4   it had earlier in the discussion, when the decision made in block 702 was to choose the

5   master node to function as portal-removal node. Thus, the algorithmic process now

6   moves to that same point, namely to block 707, and the above-described process for

7   blocks 707 through 709 including the master to node handshake of Fig. 8 are repeated. .

8   In that handshake, to be discussed in detail in connection with Fig. 8, contents of each

9   node's DDB is verified as being up-to-date or is otherwise is brought up-to-date resulting

10   in consistent or identical contents of all DDBs in the network.

11

12   **Figures 8 and 8A – Master to Node Handshake**

13   Fig. 8 is a flowchart depicting an algorithm executed by operation of

14   embodiments of the present invention in performing a master to node handshake. For

15   situations wherein a master node has need to send substantive information to its

16   participating nodes, this handshake is used with each such participating node prior to the

17   master sending that information to that participating node. For example, substantive

18   information such as the "configuration update" performed in the last step in the

19   algorithmic process from either Fig. 6 (adding a node, block 606) or Fig. 7 (removing a

20   node, block 709) is subjected to this handshake by being input to tab "A" in Fig. 8.

21

22   In query block 801, for each node in the configuration, a decision or

23   determination is made regarding whether or not the master node's IP address in the

1   configuration update (for either a node added or node removed update) matches the

2   purported master node's IP address stored in each node's DDB, as verification of the

3   master's identity. Verification is important because participating nodes in a configuration

4   must receive commands and updates from only their one true master node. Without

5   verification a problem could arise. For example, if a master node fails, requiring the

6   global administrator to appoint a different node as new master node (discussed in detail in

7   connection with Fig. 12), the new master node sends out notification to all nodes in its

8   configuration that it is their new master node. All participating nodes in the

9   configuration realign themselves to the new master node without problem. But, if the

10   failed master node recovers after appointment of the new master node, and if a user or

11   different global administrator erroneously logs-in to that previously failed master, there is

12   a chance that it might be erroneously used as master in the same configuration, which has

13   to be prevented. The verification operation defined by query block 801 prevents this

14   problem from occurring. (In addition to and concurrently with this handshake, there are

15   pinging and polling activities between master node and all participating nodes in the

16   configuration that overlays this handshake. These activities are described in connection

17   with Figs. 10A/B, 11, and 12 and deal with situations of failed nodes, failed links, as well

18   as failed masters.)

19

20   Returning to operation of block 801, if there is an IP address mismatch whereby

21   the answer is "no" for any particular participating node in the configuration, then in block

22   805 the change or configuration update is not recognized by that particular node, the

23   event is not logged into the DDB of that particular node, and the handshake is concluded.

1   In effect, this means that the particular node sends a message back to the master node

2   advising it that it is not the particular node's master (in such case the likelihood is that the

3   particular node is participating in a different domain under a different master).

4   Thereupon, the master node removes the IP address of that particular node from the

5   master node's DDB and sends the change to all other nodes in the configuration.

6   However, if the answer is "yes", the process then moves to query block 802 where

7   version numbers are compared.   For each participating node in the configuration, its

8   DDB version number *before* the configuration update is compared with the master node's

9   DDB version number *before* the update. This step ensures that each participating node's

10   DDB contents were the same as the master node's DDB contents before the update.

11

12   On the one hand, if the version numbers are the same whereby the answer to the

13   query in block 802 is "yes" then in block 804 only the update information (only the IP

14   address of the node being added or the node being removed) is accepted into that

15   participating node's DDB, and the handshake concludes. On the other hand, if the

16   answer is "no" then the version numbers do not match which means that contents of their

17   respective DDBs do not match. There is no readily available technique to determine to

18   what extent any such DDB may be out-of-date. Accordingly, for the situation of a

19   version number mismatch, the participating node flushes its contents and accepts the

20   entire contents of the master node's DDB including both updated configuration

21   information about the added or removed node and a changed new master node version

22   number. Thereafter, the handshake concludes.

23

1   Referring next to Fig. 8A, a flowchart is depicted of an algorithm executed by

2   alternative embodiments of the present invention in performing a master to node

3   handshake. The flowchart is the same as that in Fig. 8 except for the addition of two

4   blocks, 806 and 807. If there is a mismatch between a master node's IP address as

5   reflected in an update message from that master node and another master node's IP

6   address as stored in the DDB of any participating node in the configuration, then the

7   algorithmic process moves to query block 806. In query block 806 a determination is

8   made regarding whether or not the other master node's IP address stored in that

9   participating node's DDB is null (invalid). On the one hand, if "no", not null (meaning

10   valid), then such address is in conflict with the master node IP address in the update

11   message, whereby the update/change is not recognized or accepted as reflected in block

12   805, and the handshake is concluded. On the other hand, if "yes", meaning that such

13   address is null, then that nulled or invalidated address had earlier been made ineffective

14   and cannot possibly be in conflict with the master node IP address in the update message.

15   But the nulled address does indicate that such participating node's DDB contents may be

16   out-of-date. Accordingly, in block 807 such participating node flushes the contents of its

17   DDB, and updates its DDB with the new master name and a copy of all contents stored in

18   the master node's DDB, including both the update message and master node DDB

19   version number. Thereafter the handshake concludes.

20

21   **Figure 9 – Node to Master Handshake**

22   Fig. 9 is a flowchart depicting an algorithm executed by operation of

23   embodiments of the present invention in performing a node to master handshake. For

1   situations wherein participating nodes in a configuration have need to send substantive

2   information to their master node, this handshake is used for each such node prior to its

3   sending that information. For example, this handshake is used when adding a node to a

4   configuration through a participating portal-added node (see block 608 in Fig. 6) and

5   when removing a node from a configuration through a participating portal-removal node

6   (see block 705 in Fig. 7).

7

8   In block 901, an inquiring node having need to send substantive information to its

9   master gets the name and address of its presumed master stored in its own, local, DDB.

10   In decision block 902 that node inquires of the presumed master if it is that node's

11   master. If the presumed master node responds "yes", the handshake is concluded. But, if

12   the presumed master node responds "no", then the algorithmic process moves to another

13   decision block, block 903.

14

15   In block 903 the inquiring node inquires if the presumed master knows who is the

16   new master. If the answer is "yes", where the presumed master knows the identity of the

17   new master, the algorithmic process moves to block 904 wherein the presumed master

18   provides the name and IP address of the new master to the inquiring node after which the

19   handshake is concluded. But, if the answer is "no" the algorithmic process moves to yet

20   another decision block, block 905.

21

22   In block 905, which is needed when the presumed master does not know who is

23   the new master, a decision needs to be made about asking the global administrator to

1   configure a new master node by use of the select master dialog box in the GUI (discussed

2   in connection with Fig 13). If the answer to the query in block 905 is "no", the inquiring

3   node inquires again in block 903 if the presumed master knows who is the new master –

4   it is possible that the presumed master may have learned of the new master's identity.

5   For example, the presumed master may have been a master that failed and was earlier

6   replaced with a new master by the global administrator. If the presumed (failed) master

7   was not brought up-to-date by the time of occurrence of the first inquiry by the inquiring

8   node, but is brought up-to-date at a later time whereby identity of the new master is then

9   made known to the presumed master, then inquiry of the presumed master (in block 903)

10   after that later time will result in provision of the new master's identity. This iterative

11   loop or iteration means thus operates until either block 904 is utilized as described above,

12   or until the decision in block 905 is "yes" – to ask the global administrator to configure a

13   new master node by way of the GUI. If "yes", the algorithmic process moves to block

14   906 wherein the global administrator configures new master node information using the

15   GUI which results in a new master for the inquiring node and the handshake is

16   concluded.

17

18   **Figure 10A – Failed Node – Master Unknown**

19   Fig. 10A is a flowchart depicting an algorithm executed by operation of

20   embodiments of the present invention in responding to a failed node that does not know

21   which node is its master node. This situation can arise in a first scenario where the node

22   failure occurred while the node was in the process of being added to a configuration. In

23   this scenario the DDB in the node being added had not been updated with any master

1   node DDB information including the IP address of the master – thus the master is

2   unknown to this failed node. This situation can also arise in a second scenario where the

3   master node - for the configuration in which the failed node had been a participating node

4   - was replaced while the failed node was inoperative. In this second scenario the DDB in

5   the failed node had also not been updated with any (new) master node DDB information

6   including the new IP address of the new master node – thus the new master is unknown

7   to this failed node.

8

9   However, the failed node is known to the master node in either scenario. In the

10   first scenario, the master has the IP address of the node being added because it was being

11   added either through the master itself which directly provides the master with the node's

12   IP address, or through a portal-added node which forwarded the IP address of the node

13   being added to the master in the normal course of operation, earlier discussed. In the

14   second scenario the new master has the IP address of the failed node because the new

15   master inherits, or gets a copy of, content of the DDB of the replaced master, including IP

16   addresses of all participating nodes including the IP address of the failed node. In the

17   first scenario, the master sends the personified pinging message noted in connection with

18   Fig. 5: "I am the master for you" to the new node being added unaware that it has failed

19   and awaits the expected acknowledgment from that node. In the second scenario, the

20   master sends that message to all nodes in the configuration unaware that any have failed

21   and awaits the expected acknowledgment from each of its participating nodes. If the

22   expected acknowledgment is not received by the master from the node being added in the

23   first scenario or from any particular participating node in the second scenario, it shall

1   continue – at regular intervals - to ping that node from which such acknowledgment was

2   expected.

3

4   Referring-back to Fig. 4A momentarily, repetitive pinging messages 406, 407,

5   and 408 are shown emanating from master node 6 and directed to nodes 5, 7, and 8

6   respectively. Since master node 6 will not repetitively ping an operative node, repetitive

7   pinging of nodes 5, 7, and 8 would occur only if each node failed while being added to

8   the network or if each node was in a failed state when master node 6 became the new

9   master for nodes 5, 7, and 8. If, in fact, these three nodes failed while being added to the

10   configuration, with failure times that overlap each other, pinging of each node would

11   occur but not simultaneously with pinging either of the other nodes, since the nodes were

12   not being added simultaneously. On the other hand, if these three nodes had been in

13   failure at the time when node six was made their master, then the pinging of the three

14   nodes would occur substantially simultaneously, because master node 6 would discover

15   each of their failed conditions substantially simultaneously. In either case, in the best

16   mode now known for practicing the present invention, pinging intervals are

17   approximately five minutes. At the end of each of the intervals a single ping or a ping

18   burst can occur. A non-response alerts the master or the new master node to the fact that

19   the failed node has not recovered. These failure scenarios are handled by the algorithmic

20   process depicted in Fig. 10A and described as follows.

21

22   In Fig. 10A, a particular node fails in block 1001. In decision block 1002 the

23   question is asked: did the node fail while being added to the configuration? If "no" the

1   algorithmic process moves to block 1003 where the question is asked: was the master

2   node replaced during time of failure? If "no" the master is thus known to the failed node,

3   as is reflected in block 1007, and the instant algorithmic process is not applicable. The

4   algorithmic process moves away to tab "B" of Fig. 10B which deals with a different

5   circumstance - node failure when the master node is known to the failed node.

6

7   Returning to both query blocks 1002 and 1003, if the answer to the questions

8   posed in either or both these blocks is "yes", the master node is unknown to the failed

9   node as shown in block 1004. The algorithmic process moves then to block 1005, where

10 the master node repetitively pings the failed node at predetermined intervals until the

11 failed node recovers and sends a recovery signal to the master. As noted above, one of

12 such messages, or alternatively bursts of such messages, can be sent at the end of each of

13 those intervals. The recovery signal is the first of the "polling" signals to be received by

14 the master from the failed node after it recovers, and polling shall be discussed in

15 connection with Figs. 10B and 11 herein. The algorithmic process moves next to block

16 1006 in which the master node and the failed node use appropriate handshaking as

17 depicted in Figs. 8 and 9, discussed above, during which the master node updates the

18 failed node's DDB and this algorithmic process concludes.

19

20   **Figure 10B – Failed Node – Master Known**

21   Fig. 10B is a flowchart depicting an algorithm executed by operation of

22 embodiments of the present invention in responding to a failed node that does know

23 which node is its master node. In block 1008 each node in the configuration polls its

1   master at regular intervals for a DDB version check. Each node thus regularly requests of

2   its master a return message including the master's current DDB version number so that

3   the node can compare it against its version number. If the version numbers match, then

4   the node is reassured that its DDB's contents match the master's DDB contents. In the

5   best mode now known for practicing embodiments of the present invention, the regular

6   interval between polling inquiries is approximately ten minutes for each node.

7

8   Referring-back once more to Fig. 4A, nodes 5, 7, and 8 are shown polling master

9   node 6 by way of polling inquiries 403, 404, and 405 respectively. These polling

10   inquiries are not designed to occur simultaneously but must be made at the predetermined

11   interval. Thus, in a large network under normal operating conditions the master node

12   might receive such polling inquiries virtually continuously from its various participating

13   nodes. Polling at the predetermined interval by each node in a configuration of its master

14   is a continuous operation, to be contrasted with pinging by the master of nodes only

15   under certain conditions including certain failure scenarios which is not a continuous

16   operation, as discussed above. It should be understood that polling inquiries 403, 404

17   and 405 would be successful communications to master node 6 only if nodes 5, 7, and 8

18   were in fact operative and if their respective links to their master were also operative.

19   Although both repetitive pinging messages and continuous polling inquiries are both

20   shown together in Fig. 4A, it should be understood that in the earlier example where

21   nodes 5, 7, and 8 had failed and repetitive pinging messages occur there could be no

22   polling inquiries at that time – the drawing shows both sets of communications in the

23   interest of thorough explanation.

1

2   Returning to Fig. 10B, the algorithmic process moves to block 1009 wherein a

3   particular node in the configuration fails. This failed node did not fail while being added

4   to the configuration nor was its master replaced during the time of its failure, but it failed

5   under other circumstances. This state in the algorithmic process is the same as that

6   reflected in block 1007 in Fig 10A which is therefore connected at tab "B" along with the

7   flowchart output of block 1009 into block 1010. In block 1010, after the particular node

8   recovers, its next successive polling inquiry of the master is received by the master. In

9   block 1011 responsive to the received recovery signal which is the first polling message

10   after node recovery, the master sends its current DDB version number to the now-

11   recovered particular node. In decision block 1012 the query is posed: does the particular

12   node's DDB require updating? On the one hand, if the answer is "no", the algorithmic

13   process is concluded. Updating is not necessary if the master's current DDB version

14   number matches the version number of the particular node. This indicates that the DDB

15   contents were not changed in the master node's DDB during the time of failure of the

16   particular node. On the other hand, if the answer is "yes", the version numbers do not

17   match reflecting a change to the contents of the master node's DDB, and the algorithmic

18   process moves to block 1013. In block 1013 appropriate handshaking as depicted in Figs.

19   8 and 9 are undertaken, after which, or as part of which, the particular node's DDB is

20   updated to match that of the master node and the algorithmic process is concluded.

21

22   **Figure 11 – Failed Network Link**

1   Fig. 11 is a flowchart depicting an algorithm executed by operation of

2   embodiments of the present invention in responding to a failed network link between a

3   particular participating node in the configuration and its master node. For example, in

4   Fig. 1, storage system 104 can contain a plurality of subnets or domains (not shown) each

5   with its own master, and any link could fail between any node and its master in any of

6   those subnets within such storage system. Such a link could be a hardware, software, or

7   hardware/software hybrid link. For another example, a link could also encompass

8   wireless communication between a network master node and participating nodes in its

9   configuration .

10

11   In block 1101 each participating node polls the master node at regular intervals on

12   a continuous basis to compare the version number of its DDB with the master node's

13   version number. In block 1102 a network link between the master node and a particular

14   node fails. The particular node and the master node are otherwise fully operative. While

15   the link has failed, the particular node still attempts to continuously poll its master to

16   compare version numbers, but because of the failed link the master does not receive such

17   polling and accordingly does not respond. In block 1103 after the failed link recovers,

18   the next successive polling inquiry made by the particular node of the master after

19   recovery is received by the master. In block 1104, responsive to receiving that next

20   successive polling inquiry over the now-recovered network link, the master node sends

21   its current DDB version number to the particular node over that link. In query block

22   1105 a version number comparison is made – the question is posed: does the particular

23   node's DDB require updating? On the one hand, if the answer is "no", the algorithmic

1   process stops. On the other hand, if the answer is "yes", then some event occurred during

2   the time of the failed network link to cause the master to change the contents of its DDB

3   and change its version number accordingly. The algorithmic process then moves to block

4   1106 wherein appropriate handshaking is undertaken as depicted in Figs. 8 and 9 and that

5   particular node's DDB is updated to match that of its master node, after which the

6   algorithmic process concludes.

7

8   **Figure 12 – Failed Master Node**

9   Fig. 12 is a flowchart depicting an algorithm executed by operation of

10   embodiments of the present invention in responding to a failed master node. Consider an

11   operating network configuration, having a master node with its plurality of participating

12   nodes, initially all nodes being operational. In block 1201 the master node fails for some

13   reason. In block 1202, at some period of time after the master node's failure, the global

14   administrator by way of the GUI attempts to change the configuration in some manner

15   (e.g. adding or removing a node) but cannot do so. The global administrator instead

16   receives an error response advising that no change to the configuration is possible since

17   the master has failed. In block 1203, using the select master dialog in the GUI (Fig. 13),

18   the global administrator appoints a different node in the configuration to be the new

19   master for the configuration, which is permitted.

20

21   Until a change is attempted by the global administrator, participating nodes in the

22   configuration continue to align themselves to the failed master, polling such failed master

23   at regular intervals, but not receiving any version number check responses. The DDB

1 information in each of those participating nodes is thus frozen until a new master is

2 appointed. During the time of failure of the former master, which runs from the

3 occurrence of the failure until discovery of the failure by the global administrator and

4 appointment of a new master, other network configuration changes can also occur

5 including, for example, one or more participating nodes can fail and/or one or more

6 network links can fail.

7

8 The algorithmic process moves from block 1203 to block 1204. In block 1204 the

9 new master sends out its "I am the master for you" message to each node whose IP

10 address it has in its DDB which was obtained from the DDB of the failed former master

11 node, including sending it to the failed former master node. As noted earlier in

12 connection with a discussion about the pinging operation, the new master node then

13 expects an acknowledgment from each of the nodes to which it sent this message, and if

14 there are no acknowledgments from certain nodes (such as the failed former master node)

15 then the new master node knows to repetitively ping these certain nodes at the

16 predetermined pinging interval. Thus, in block 1205 the new master node sends out this

17 same message over and over again as a repetitive ping at predetermined intervals to the

18 former failed master node and to any other possibly failed nodes in its configuration

19 which are now known to the new master because of no acknowledgments received back

20 from those possibly failed nodes. In block 1206, when the failed former master node

21 initially recovers it mistakenly believes itself to still be master for this configuration until

22 it receives the next successive ping message directed to it by the new master advising it

23 that: "I am the master for you". The now-recovered former master node then sends the

1   appropriate acknowledgment to the new master whereby the new master stops pinging it.

2   The now-recovered former master node next undergoes appropriate handshaking by way

3   of Figs. 8 and 9, during which the new master updates the DDB of the now-recovered

4   former master node which is now demoted to just another participating and operative

5   node of the configuration with its DDB contents matching that of the new master node.

6   If there are other failed nodes, upon eventual recovery of each one of them it

7   acknowledges the new master which stops the pinging by the master to it, and such

8   recovered node goes through the handshaking process in Figs. 8 and 9 to receive its

9   appropriate DDB contents update.

10

11   If links failed during the time of failure of the failed former master node, the new

12   master will attempt to repetitively ping the nodes connected to those failed links because

13   such nodes cannot receive the initial "I am the master for you" message over failed links

14   and therefore cannot acknowledge it, thus appearing to be failed nodes. Nor can such

15   nodes communicate their polling inquiries to the new master over failed links. In block

16   1207, upon recovery of those failed links, nodes connected from those now-recovered

17   links also are brought up-to-date, but in accordance with the algorithm depicted in Fig. 11

18   which brings into account the handshaking operations of Figs. 8 and 9. Thereupon the

19   algorithmic process of Fig. 12 is concluded.

20

21   **Figure 13 – Select Master Dialog Box**

22   Figure 13 is a representation of a typical computer terminal screenshot associated

23   with operation of certain embodiments of the present invention employing a graphical

1   user interface (GUI) with a computer network including storage systems. A GUI could

2   be employed within client or management workstation or user interface 100 in Fig. 1.

3   The screenshot depicts a dialog box of the type which would enable a global

4   administrator (privileged user) to control master node selection.

5

6   With regard to screen layout, in the upper left hand corner of the Figure, a "Select

7   Master" title in the toolbar is shown and in the upper right hand corner in that toolbar an

8   exit button "X" is shown and used to exit the dialog box. Under the heading "Master

9   Node IP Address" there is shown an editable field containing, in this example, a specific

10 IP address: 128.221.34.187. The global administrator can appoint or change master

11 nodes by editing this field, described below. There is a "Master Candidate Nodes"

12 section in the screenshot. This section displays nodes which are participating, configured

13 nodes in the domain and are thus available as candidates from which one can be selected

14 or elected by the global administrator as master. In this section, three categories of

15 information are provided: "Node Information", "System Name" and Provider

16 Information".

17   • Node Information includes IP address and system identifier of a particular

18   node or system. As noted earlier, an entire storage system can be treated in

19   object-oriented software as an object or node and thus can have an IP address

20   (or possibly more than one address) associated with it. The system identifier

21   is constructed from the last two segments of the IP address of that system (e.g.

22   128.221.34.187 is the IP address corresponding to system identifier

23   CPC34187).

1   • System Name is the name assigned to the system or node and is just a label or

2   tag.

3   • Provider Information includes name and version number of providers installed

4   in the system. A provider is a functional software module. This version

5   number is not the same as version number of the master node and its

6   participating nodes earlier discussed, but is version number of the provider

7   software.

8   There are three entries shown in the Master Candidate Nodes section, having system

9   names "Array 3", "Array 2" and "Array 1", with associated Node Information and

10   Provider Information. In the lower right corner of the Figure are four control buttons:

11   "OK", "Apply", "Cancel", and "Help".

12

13   With regard to operation, the global administrator can point the cursor and mouse-

14   click on Apply (can hit Apply button) to make a node selected from this group of three

15   nodes appear in the editable field above and become new master for this configuration.

16   In this example, the second candidate in the Master Candidate Nodes section

17   corresponding to Array 2 is selected as master node, as is reflected in the editable field.

18   The OK button performs the same action as the Apply button, but also closes the dialog

19   box. The Cancel button closes the dialog box without saving the global administrator's

20   changes thereby maintaining the status quo. And, the "Help" button launches a help

21   menu for this Select Master dialog box.

22

1   However, before any of this operation can be performed, the global administrator

2   first has to bring up the dialog box. And, before the dialog box can be brought up, the

3   global administrator first has to bring up a web browser on his/her terminal screen and

4   type into the Uniform Resource Locator (URL) slot the IP address of a storage system

5   chosen by the global administrator (in this example, one of the IP addresses for Array 1,

6   2, or 3). Such storage system may be part of a small or large domain or network of

7   storage systems and possibly other nodes. In this example the global administrator could

8   have typed-in the IP address of any of the three arrays shown and the same substantive

9   information would have been displayed because the DDB in each of the three storage

10   systems contains the same directory data. However, the information may be displayed in

11   a different order as a function of which IP address was typed in. By accessing a system

12   or node in this manner, a framework software is brought into operation, which provides

13   the framework or foundation of the terminal screen display. With reference to one of the

14   incorporated-by-reference patent applications, "Plug and Play Interface for User

15   Actions", Desai et al, U.S. Serial No. 09/916102, filed July 26, 2001, a dialog box of this

16   kind can be created by a software module that plugs into such framework software as

17   described in that patent application. However, after framework software comes up, a

18   security operation involving password access is next undertaken to ensure that only a user

19   with proper password (presumably global administrator) has access to the Select Master

20   dialog box. If the proper password is supplied, the framework software thereafter offers a

21   menu item "Select Master". And, if that menu item is pointed-to and clicked-on by the

22   global administrator, or other authorized user, then, this dialog box is finally launched.

23

1   The act of launching this dialog box causes reading of directory information from

2   the DDB in the storage system node chosen by the global administrator. In this example,

3   that DDB contains node, system name, and provider information of Array 3, Array 2 and

4   Array 1 respectively, and displays it as shown. This information shows that this DDB

5   has three storage arrays - three IP addresses in this case - and that the chosen storage

6   system is thus participating in a domain or configuration that has three nodes. In the case

7   where a storage array has two IP addresses associated with it, the same system name will

8   appear on the screen twice, reflecting both IP addresses on two separate rows.

9

10   Any one of these three nodes (if all are operative) can be selected by the global

11   administrator to be master node by merely highlighting that selection in the Master Node

12   Candidates section and hitting the OK or Apply button. This action will move the IP

13   address corresponding to that selection to the Master Node IP Address editable field and

14   replace the IP address of any node that might have been in that field. That replaced

15   master node is then demoted to being a participating node of the configuration.

16   (Referring back to Fig. 4B, the "M" identifier is moved from Node 1 to either Nodes 2, 3,

17   or 4 in that domain.) If one of the listed nodes had failed, for example Array 1, and is

18   thereafter selected to be master node, then after highlighting Array 1 and hitting the

19   Apply button the user interface returns an error signal indicating that CPC 3463 was not

20   able to be contacted. Another node will then have to be selected. As noted, in this

21   example, the global administrator selected array 2 and its IP address appears in the

22   editable field at the upper portion of the screenshot. After the global administrator is

1   finished with his/her master node selection, the dialog box can be closed out by hitting

2   the X in the upper right corner.

3

4   **Figure 14 – Configure Domain Dialog Box**

5   Fig. 14 is a representation of a typical computer terminal screenshot associated

6   with operation of the same embodiments of the present invention that produced the dialog

7   of Fig. 13. Fig. 14 depicts a dialog box of the type which would enable a global

8   administrator user to configure or reconfigure a domain. In the main toolbar section,

9   "Configure Domain" is shown as the title of the dialog box . Also, in the main toolbar,

10 right hand side, an exit button "X" is shown and is used to exit the dialog box by clicking

11 on it.

12

13   With regard to screen layout, there are three sections displayed: "Domain Name"

14 near the top, "Scan Subnets" below that, and "Select Systems" at the bottom half of the

15 screenshot.

16   •   In Domain Name section its editable field shows name of domain of the

17   system selected by the global administrator. The term "DefaultDomain" is

18   shown in the field, and a default domain is automatically selected if not

19   overridden by a different selection made by the global administrator. There is

20   a "Change" button located to the right of this field.

21   •   In Scan Subnets section there are two fields, "Subnets To Add" which is an

22   edit control and "Subnets To Scan" which is a list control. There are also four

1   buttons shown: "A", "Scan", "Stop Scan" and "Clear". And there is a scan

2   progress bar.

3   • In Select Systems section there is an editable field entitled "IP Address of

4   System" operated upon by a "B" button to its right; an "Available Systems"

5   field operated upon by "C" and "D" buttons to its right, and a "Selected

6   Systems" field operated upon by three "B", "C", and "D" buttons to its left.

7   In addition, there is a "Clear" button at bottom of this section, as well as scroll

8   bars shown at bottom of both Available Systems and Selected Systems fields.

9   At bottom of the screenshot are four buttons: "OK", "Apply" "Cancel" and "Help".

10

11   With regard to operation, this dialog is brought up in the same manner as

12   described earlier in connection with the dialog of Fig. 13. In this case, there is a

13   "Configure Domain" menu item presented which the Global Administrator clicks on and

14   which launches this dialog box. When this dialog is launched, the name of the system or

15   node pointed to by way of inserting its IP address into the URL slot appears in the

16   Domain Name field in this dialog. If the global administrator wants to change this name

17   he/she hits the Change button which brings up another dialog (not shown) with an

18   editable field allowing the typing-in of a different name. The domain name is visible

19   from all participating systems in that domain which means that this same name would

20   appear in the Domain Name field regardless of which participating system's IP address

21   was put into the URL slot by the global administrator. Two domains can be given the

22   same name without malfunction, (they could all have the default domain name) but this

1   could lead to confusion. In any event, any domain name must be in alphanumeric form

2   only, and non-alphanumeric symbols are not permitted.

3

4   In the Subnets to Add edit control the global administrator types in the address of

5   a subnet which he/she wishes to have scanned to discover any available systems (systems

6   unaligned to any master node) that are also compatible with other nodes in the domain

7   which the global administrator is in the process of configuring or re-configuring. For

8   example, nodes or storage systems or servers which fall into the category of Common

9   Information Model Object Manager (CIMOM) systems or servers are mutually

10   compatible. Note that this address is not a full four-segment IP address, but is a subnet

11   address or identifier (ID) having only three segments. The "A" button located to the right

12   of the Subnets To Add edit control is used to for moving contents of that edit control to

13   the Subnets to Scan list control located to the right of the button, thereby populating that

14   list control field which is initially empty when the dialog is brought up. A populated list

15   control field thus reflects certain subnets which the global administrator wishes to scan.

16   As can be seen, in this example subnet address "10.14.12" is the last subnet that was

17   added to the Subnets to Scan field, as it is located at bottom of three subnet addresses

18   shown in that field. The global administrator uses the screen cursor to highlight one or

19   more subnets listed and hits the Scan button to the right of the list control to start a

20   discovery operation on those highlighted subnets. The discovery operation will cause a

21   display in the progress bar to show percentage of completion of scan operation. The Scan

22   button is disabled from time that scanning is started until it is finished. The Clear button

23   is used for clearing contents of list control, this button being enabled only if the global

1   administrator selects (highlights) a subnet listed in the list control. The Clear button is

2   disabled and a subnet cannot be highlighted in the list control field when scan operation

3   is in progress. The Stop Scan button is used to stop scan operation which was started by

4   hitting the Scan button, and will be enabled only if there is a scan in progress.

5

6   After a scanning operation has completed, where progress bar shows 100%, IP

7   addresses of any discovered nodes that are unaligned to another master node and which

8   are compatible with nodes in the domain under configuration are automatically sent to the

9   Available Systems field in the Select Systems section of the dialog. In other words, these

10   particular discovered nodes populate the Available Systems field which has sub headings

11   of "System" (storage system name) and "Node Info" (storage system IP address). In the

12   example shown, in the Subnets To Scan field, the subnet having address 128.221.34

13   contained an unaligned and compatible node with IP address 128.221.34.80 with system

14   name Cadsys, and this is the first entry in the Available Systems window. Likewise, the

15   subnet having address 128.221.42 contained three unaligned and compatible nodes:

16   128.221.42.64; 128.221.42.80; and, 128.221.42. 122 which are three entries under Node

17   Info in the Available Systems field. Finally, the last subnet scanned having address

18   10.14.12 contained an unaligned and compatible node with IP address of 10.14.12.105

19   which is the last entry shown in the Available Systems field.

20

21   The global administrator now has a choice of selecting any or all of these

22   available systems for addition to the domain being configured. For example, to add the

23   node named Cadsys the global administrator would highlight this entry in the Available

1   Systems field and hit the C button. This action would move that Cadsys entry from the

2   Available Systems field which would thus be decreased by one entry, to the Selected

3   Systems field to the right of the button which would thus be increased by one entry.

4   Then the global administrator hits the Apply or OK button to cause the master node to

5   replicate the node added change in the DDBs of all nodes in the domain and thereby keep

6   the DDBs consistent throughout the domain.

7

8   In the Selected Systems field there are shown three nodes having the same IP

9   addresses that were shown in Fig. 13. These addresses appear in this field when this

10   Configure Domain dialog is brought up, which means that the domain under

11   configuration in Fig. 14 is the same domain for which the global administrator selected

12   Array 2 having IP address 128.221.34.187 as master node in Fig. 13. In other words, in

13   this illustrative example, the three nodes appearing in the Selected Systems field had been

14   earlier selected for this domain and can thus be considered as "pre-selected" since they

15   existed in the field prior to bringing up the dialog shown in Fig. 14. In accordance with

16   our example, this entry of three nodes in the Selected Systems field can be increased by

17   the addition of the Cadsys system and any others selected by the global administrator.

18

19   In the reverse operation, the population of the Selected Systems field can be

20   reduced by the global administrator highlighting an entry in the Selected Systems field

21   and hitting button "D" which moves that highlighted entry into the Available Systems

22   field. Then the global administrator hits the Apply button which causes the DDB in the

23   removed node to destruct (to discard its contents), thereby unaligning its DDB from

1   alignment to master node cpc34187. This unalignment makes the removed node

2   available for use by other global administrators. Hitting the Apply button also causes the

3   master node to replicate this node removal change in all the remaining DDBs in the other

4   nodes of this domain and thereby keep the DDBs consistent throughout the domain. The

5   "Clear" button is used to clear or empty contents of the Available Systems field to make

6   it open for repopulation by newly-discovered available systems derived from the

7   scanning operation in the Scan Subnets section of the dialog. The cleared systems which

8   are thus removed from view from the Available Systems field are otherwise unaffected

9   and their DDBs remain intact. Scroll bars at bottom of both the Available Systems field

10   and the Selected Systems field, also operable through left and right pointing arrow

11   buttons at lower left and right corners of the fields, permit vertical scrolling of entries in

12   both fields.

13

14   In the editable "IP Address of System" field located above the Available Systems

15   field the IP address 128.221.34.16 is shown. This is an IP address of a node about which

16   the global administrator has prior knowledge, and knows is both unaligned to any other

17   master and compatible with nodes of this domain under configuration. The global

18   administrator has thus typed-in this IP address into this editable field. By clicking on the

19   "B" button, this IP address is moved into the selected Systems field and the OK or Apply

20   button is then hit to add the node with this IP address to the configuration as described

21   above. If the global administrator later wants to remove it, highlighting it and hitting the

22   D button moves it over to the Available Systems field, in accordance with operation

23   discussed above.

1

2   It should be understood that embodiments of the present invention relating to the

3   GUI discussion of Figs. 13 and 14 utilize JAVA® object oriented software, including the

4   framework software discussed above and further discussed in the incorporated-by-

5   reference patent application cited in discussion of Fig. 13A Software of the present

6   invention operates on, through, and/or with a client workstation such as a Windows NT

7   or UNIX® workstation to accomplish various GUI functions and operations, including

8   those discussed in connection with both Figs. 13 and 14.

9

10

11   **Figures 4A and 8 – Marriage of Two Subnets**

12   With the GUI operational description of Figs. 13 and 14 in mind, and referring

13   back to Fig. 4A and Fig. 8, consider the marriage of two subnets. In Fig. 4A, where two

14   domains or subnets are shown operating independently, each subnet is shown to have its

15   own master. The DDB of each master node contains an IP address for each node in its

16   own subnet. These two subnets can be physically linked or joined together (married) by

17   the global administrator operating through the GUI, so that any node from either domain

18   can communicate with any node from the other domain. In this married state, although

19   communicatively linked, the nodes from each subnet behave in a manner consistent with

20   their prior independent subnet status.

21

22   Using Fig. 4A as an example, in a subnet marriage, the two masters (nodes 1 and

23   6) remain within their respective domains, and the domain boundaries around domains

1   401 and 402 remain intact. However, there will be an operative coupling, a

2   communication link or bus (not shown), that connects nodes 1 – 4 inclusive from domain

3   401 to nodes 5 - 8 inclusive from domain 402. For example, node 3 could talk to node 8,

4   etc. Accordingly, if an IP address from one of the nodes from one of the subnets, for

5   example node 2 from domain 1, is inserted into a web browser's URL slot on the GUI

6   terminal screen, and a scan operation of this marriage of two subnets is performed, then

7   nodes from both subnets will be discovered. (This instant scan is similar to, but different

8   from, that shown in Fig. 13. The scan of Fig. 13 was for the purpose of discovering

9   available nodes outside of a domain with which to configure or re-configure that domain,

10   but this instant scan is to discover nodes currently existing within a domain and is

11   performed through another dialog, not shown).

12

13   The result of this instant scan will cause node 2 to initiate an update to its master

14   node 1, advising it of new nodes in the domain. Master node 1 will update its DDB with

15   all node information from all nodes in both subnets. But, when master node 1 propagates

16   its update to all nodes from both subnets, those nodes which comprise domain 402 reject

17   the update. Those nodes have their own master node, master node 6. Because of the

18   master to node handshake, nodes 5, 6, 7, and 8 will reject this update request since the IP

19   address of master node 6 differs from the IP address of master node 1 (see Fig. 8, block

20   805). The global administrator will be able to see this at the GUI by looking at an

21   eventlog. Master node 1 will mark nodes 5, 6, 7, and 8 in its DDB as remote nodes

22   within its subnet. Likewise, master node 6 will mark in its DDB nodes 1, 2, 3, and 4 as

23   remote nodes within its subnet. Thereafter, until further intervention by the global

1   administrator, both sets of nodes co-exist in a marriage of subnets while retaining their

2   individual autonomy.

3

4   This network configuration has practical utility when a link between domains is

5   useful or economically feasible, but not on a continuous basis. For one example,

6   consider the case where a startup company on a tight budget has two offices located on

7   opposite coasts of the United States, and where they have need to access each others files.

8   This business arrangement might arise when a West coast project is started at a different

9   time from the East coast project and where it is useful to keep the two projects segregated

10   except for intra-company access to each other's files. This company may lease network

11   lines that are charged at an hourly rate to connect these two subnets cross country. And,

12   with this marriage scheme, they can be disconnected each evening and reconnected each

13   morning to effect a substantial savings in leased line costs. While these subnets are

14   connected, co-workers at opposite ends of the country can read each others files at great

15   convenience to the company, but they cannot otherwise impact each other's files, keeping

16   the two projects segregated. The DDBs in the East coast's nodes are isolated from the

17   West coast's master node and vice versa.

18

19   The present embodiments are to be considered in all respects as illustrative and

20   not restrictive. The flowcharts used herein to demonstrate various aspects of the

21   invention should not be construed to limit the present invention to any particular logic

22   flow or logic implementation. For example, the alternative embodiment of the master to

23   node handshake as depicted in Fig. 8A, shows that described logic may be combined or

1   partitioned into different logic blocks, (e.g., programs, modules, functions, or

2   subroutines) without changing the overall results or otherwise departing from the true

3   scope of the invention. The present invention may thus be embodied in many different

4   forms, including, but not limited to, computer program logic for use with any kind of

5   processor, programmable logic for use with any kind of programmable logic device,

6   discrete components, integrated circuitry including application specific integrated circuits

7   (ASICs), or any other means including any combination thereof. Computer program

8   logic implementing all or part of the functionality described herein may be embodied in

9   various forms, including, but not limited to, source code form, computer executable form,

10   and various intermediate forms (e.g. forms generated by an assembler, compiler, linker,

11   or locator.) Source code may include a series of computer program instructions

12   implemented in any of various programming languages for use with various operating

13   systems or operating environments. The source code may define and use various data

14   structures and communication messages. The source code may be in computer

15   executable form, or it may be in a form convertible into computer executable form. The

16   computer program may be fixed in any form either permanently or transitorily in a

17   tangible storage medium, such as a semiconductor memory device, a magnetic memory

18   device, an optical memory device, a PC card, or other memory device. The computer

19   program many be fixed in any form in a signal that is transmittable to a computer using

20   any of various communication technologies including, but not limited to, analog, digital,

21   optical, wireless, networking, and internetworking technologies. The computer program

22   may be distributed in any form as a removable storage medium with accompanying

23   printed or electronic documentation, preloaded with a computer system (e.g. on system

1   ROM or fixed disk), or distributed from a server or electronic bulletin board over the

2   communication system (e.g., the Internet or World Wide Web).

3

4   The present invention may be used in any network environment where a

5   distributed directory database is needed or utilized, such network being used for any

6   purpose including, but not limited to, computer data storage. Furthermore, although

7   embodiments of the present invention include C++ and JAVA object oriented software,

8   other software could be utilized. Therefore, the scope of the invention is indicated by the

9   appended claims rather than by the foregoing description, and all changes which come

10   within the meaning and range of equivalency of the claims are intended to be embraced

11   therein.

12

13